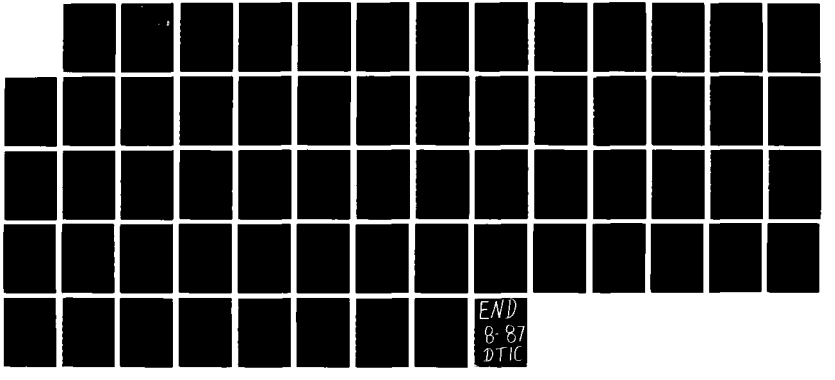
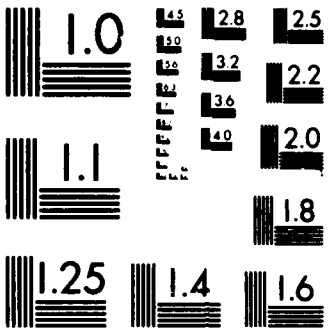
AD-A182 730 INVESTIGATION OF CURRENT STATE OF CRYTPOGRAPHY AND THEORETICAL IMPLEMENTA..(U) COLORADO STATE UNIV FORT COLLINS E A FETZNER MAY 87 1/1

UNCLASSIFIED F/G 12/9 NL





INVESTIGATION OF CURRENT STATE OF CRYTPOGRAPHY AND THEORETICAL IMPLEMENTATION OF A CRYPTOGRAPHIC SYSTEM FOR THE COMBAT SERVICE SUPPORT CONTROL SYSTEM

by

Edward A. Fetzner
First Lieutenant, United States Army
B.S., Longwood College, 1985

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from

Colorado State University
May 1987

Author: Edward A. Fetzner
Edward A. Fetzner

Approved by: Dale Grit
Dale Grit, Committee Chairman

Jacob L. Wolf
Jay Wolf, Committee Member

Nicholas Krier
Nicholas Krier, Committee Member

To whom it concerns:

The inclosed report is the written part of my Master's research in computer science at Colorado State University. I have provided you copies to maintain at your facilities. If there are any questions I can be contact at the following address:

2112 Moncrieff Street
Augusta GA 30906

Edward A. Fetzner
1Lt, Signal

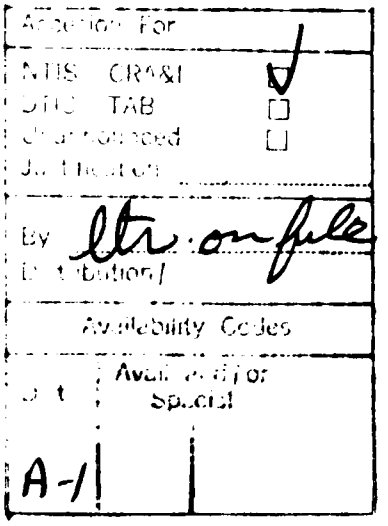


TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

The military has been researching a system that will provide commanders with a tool for organizing, directing, coordinating, and controlling the activities of military forces in order to attain an objective. The system is known as the Army Command and Control System (ACCS), which consist of five Command and Control (C2) sub-systems: Air Defense Artillery, Maneuver Control, Electronic Warfare/Intelligence, Fire Support and Combat Service Support.

The system will be used to distribute and store information from around the battle field for use in decision making. Due to the nature of a battle field environment, security mechanisms are essential. Specifically, the control over users accessing the system and, information either stored in or communicated through the system. An informal security system model for the Combat Service Support System (CSSCS) [Heng87] has been researched. The following investigation will address the communication and protective storage aspects of the informal model in closer detail.

First, several cryptographic techniques and function are compared to find a suitable cryptosystem to utilize. Following this is a discussion of a theoretical implementation and integration of the cryptosystem with CSSCS. The informal model and some basic concepts of cryptography are outlined in the next section.

PRIMER

The following information is presented to acquaint readers with basic terms and concepts of cryptography. Also, a brief overview of the Military Security system and the hardware used is supplied for the reader.

A. CRYPTOGRAPHY

Cryptography consists of two *one-to-one* functions, *encryption* and *decryption*. The encryption function maps elements of the *plaintext* (original message) onto elements of the *ciphertext* (coded message). The decrypting function would then map elements of the ciphertext onto elements of the plaintext, to recover the original message. For example let:

P be the set of plaintext elements or the domain of the function.

C be the set of ciphertext elements or the co-domain of the function.

F_E be the function which associates each element of P to a single element of C.

F_D be the function which associates each element of C with a unique element of P.

Mathematically:
$$F_E(P) = C \quad (1)$$
$$F_D(C) = P \quad (2)$$

To ensure that no ambiguities arise from deciphering, the functions must be one-to-one and onto. Thus, each enciphering function has an unique inverse and each element of the plaintext has unique image in the ciphertext. The number of

functions for a given set of plaintext elements are:

$$\frac{|C|!}{(|C| - |P|)!}$$

(| | is the total number of elements)

The E & D in equations 1 & 2 are known as *cipher keys*. Cipher keys are a subset of the total number of functions. The subset is usually substantially smaller than the whole set of functions, to avoid having cases where all plaintext-to-ciphertext correspondences are identical even when different cipher keys are used. This is known as *equivalent keys*, which could affect the usefulness of the algorithm.[Meye82; Katz77; Lemp79]

B. BLOCK CIPHER

A Block cipher encrypts and decrypts data in blocks of elements. The blocks usually consists of some number of bits or bytes from the data. The size of the block is a predetermined fixed number, chosen by the algorithm's designer. The encryption and decryption algorithm consists of transforming a block of input bits into a fixed block of output bits using a cipher key.[Meye82; Lemp79]

C. STREAM CIPHER

In a stream cipher the user defines the length of the data to be ciphered. The encryption and decryption algorithm uses a cipher key to generate an *initialize vector* or cryptographic bit-stream. The initialize vector is then

combined, using *Exclusive-Or (modular 2)* operation, with the plaintext to produce the ciphertext when encrypting. Similarly, with the ciphertext when decrypting.[Meye82; Lemp79]

D. CONVENTIONAL CIPHER ALGORITHMS

A Conventional encryption algorithm employs the same cipher key for encrypting and decrypting. This is known as a *symmetric cryptosystem.* Mathematically:

$$\text{Encipher: } f_k(P_i) = C_i$$

$$\text{Decipher: } f'_k(C_i) = P_i$$

Conventional algorithms are usually block ciphers, although they can be implemented as a stream cipher. A conventional block algorithm first splits a data block into two equal halves, R(0) & L(0). Next, R(0) is put through a function using a cipher key and added, modulo 2, to L(0) to produce R(1). L(1) is set equal to R(0):

$$R(1) = L(0) \text{ XOR } f_k(R(0)) \text{ \& } L(1) = R(0)$$

This process is continued using different cipher keys. This is known as *exercising* the function with the set of keys, K: $(k_1,k_2,\ldots,k_n)$. To decipher, either the same or different function is used with the cipher keys in reverse order than they were in enciphering, K: $(k_n,\ldots,k_2,k_1)$. Thus, the cipher key values must be known by both users. Figure 1 gives an overview of a conventional algorithm.[Meye82; Simm79]

E. PUBLIC-KEY ENCRYPTION ALGORITHMS

A public-key algorithm employs one key for enciphering, which is released to public knowledge, and a decrypting key which is kept secret. This is known as an *asymmetric cryptosystem.* A public-key algorithm must also have the following properties:

1) The public cipher key (PK) and secret cipher key (SK) must be easily and efficiently computed by the system.

2) Having the knowledge of PK can not allow SK to be computed easily. There is no requirement that knowing SK must prevent PK from being calculated easily.

3) The ciphering of the text can be performed by either using the encrypting function and SK followed by the decrypting function and PK; or use the functions and keys in reverse order:

$$D_{sk}(E_{pk}(P)) = E_{pk}(D_{sk}(P)) = P$$

The encrypting and decrypting function is based on a NP-complete problem. A NP-complete problem is one in which there does not exist a polynomial-time algorithm to solve the general problem used in the function.[Meye82; Lemp79; Simm79; Pope79]

F. DATA ENCRYPTION STANDARD ALGORITHM (DES)

DES algorithm is a conventional encryption algorithm that can either be a block or stream cipher. The original Algorithm published by the Nation Bureau of Standards describes the process using 64 bit text blocks and cipher keys.

An initial permutation splits the text block into two 32 bit blocks, L(0) & R(0). The splitting operation assigns bits of the original text to positions in either L or R:

$$L(0) = P_{58},P_{50},P_{42},P_{34},P_{26},P_{18},P_{10},P_{2},$$
$$P_{60},P_{52},P_{44},P_{36},P_{28},P_{20},P_{12},P_{4},$$
$$P_{62},P_{54},P_{46},P_{38},P_{30},P_{22},P_{14},P_{6},$$
$$P_{64},P_{56},P_{48},P_{40},P_{32},P_{24},P_{16},P_{8},$$

$$R(0) = P_{57},P_{49},P_{41},P_{33},P_{25},P_{17},P_{9},P_{1},$$
$$P_{59},P_{51},P_{43},P_{35},P_{27},P_{19},P_{11},P_{3},$$
$$P_{61},P_{53},P_{45},P_{37},P_{29},P_{21},P_{13},P_{5},$$
$$P_{63},P_{55},P_{47},P_{39},P_{31},P_{23},P_{15},P_{7},$$

Recall, that a conventional algorithm used a set of keys to exercise the function used for encrypting and decrypting. DES uses a set of 16 keys. Initially the algorithm is supplied a 64 bit cipher key. The key first has 8 bits split off, bits: 8, 16, 24, 32, 40, 48, 56 & 64. These are stripped off to ensure that there are no equivalent keys and for error detection. The remaining 56 bits are split into two 28 bit blocks, D_0 & C_0:

$$C_0 = k_{57},k_{49},k_{41},k_{33},k_{25},k_{17},k_{9},$$
$$k_1,k_{58},k_{50},k_{42},k_{34},k_{26},k_{18},$$
$$k_{10},k_2,k_{59},k_{51},k_{43},k_{35},k_{27},$$
$$k_{19},k_{11},k_3,k_{60},k_{52},k_{44},k_{36}$$

$$D_0 = k_{63},k_{55},k_{47},k_{39},k_{31},k_{23},k_{15},$$
$$k_7,k_{62},k_{54},k_{46},k_{38},k_{30},k_{22},$$
$$k_{14},k_6,k_{61},k_{53},k_{45},k_{37},k_{29},$$
$$k_{21},k_{13},k_5,k_{28},k_{20},k_{12},k_4$$

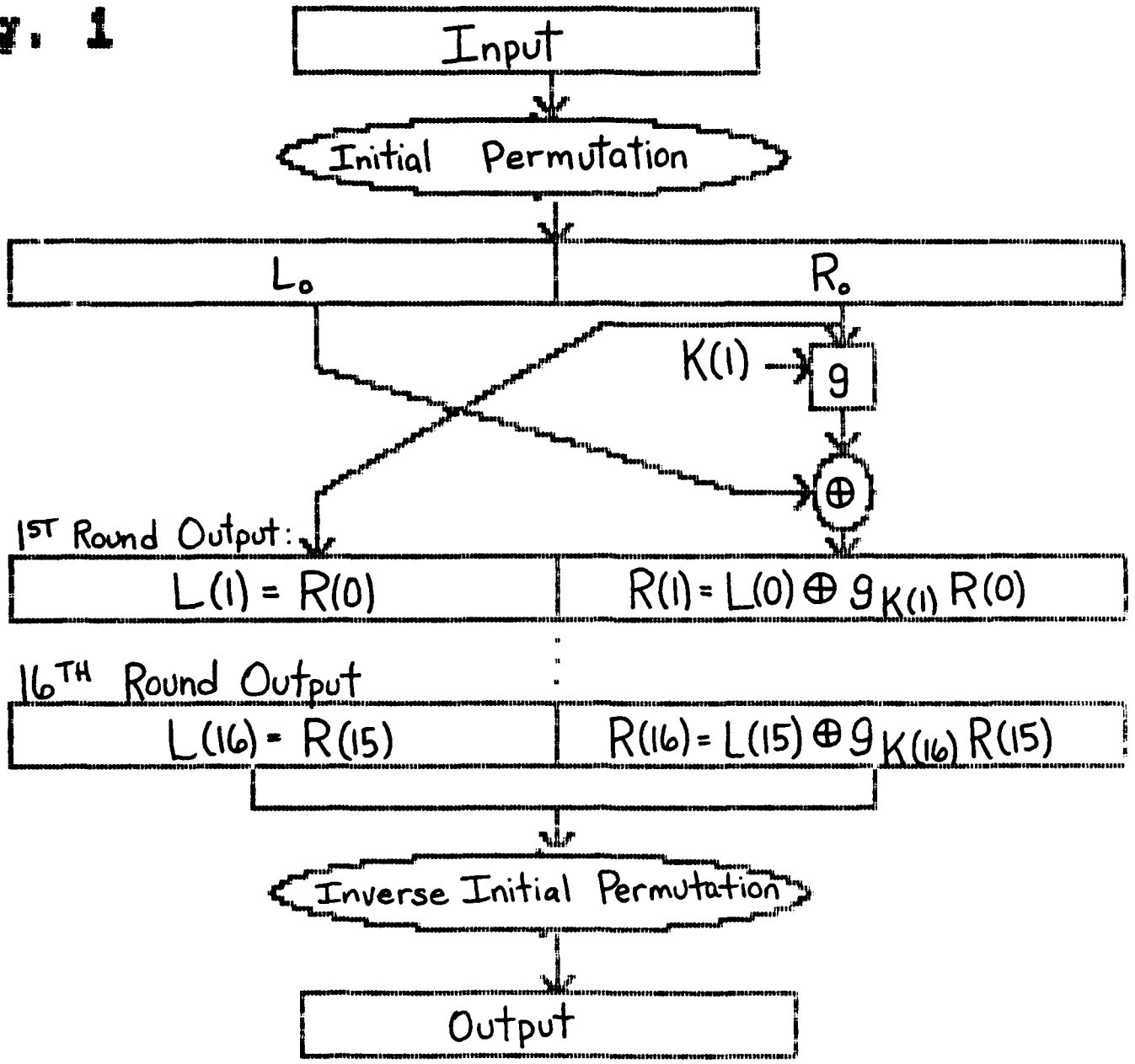
The remaining 15 keys are produced by shifting the previous round's key to the left one or two bits in a wrap around fashion. The schedule for shifting is such that $C_0 = C_{16}$ and $D_0 = D_{16}$.

Round Number:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Left Shifts:	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

The function (G_k in figure 1) first expands R(x) from 32 bits into 48 bits. This is accomplished by duplicating 16 of the original bits and reassigning all the bits to different bit locations in the 48 bit block. The 48 bit block is then added modulo 2 (bit-by-bit) to the cipher key used for a particular round. Next the resulting 48 bits are filtered through 8 substitution operations or *S-boxes*, the first 6 bits through the first S-box (S_1), the second 6 bits through S_2, etc. Each S-box is composed of a 4X16 matrix, with each row containing the numbers 0-15 (binary) in various positions. The first and last bit of the 6 bits used by a S-box determine which row of the matrix and the middle four determine which column. Thus, the 48 bits are transformed back to 32 bits after the S-boxes. Lastly, the function permutes the resulting 32 bits by reassigning bit positions and adding (modular 2) L(x) to produce R(x+1). L(x+1) equals R(x) as used in a generic conventional algorithm.

After the plaintext is split by the initial permutation and ciphered by the function, the inverse of the initial permutation is performed to produce the cipher text. Decrypting uses the same function but with the keys in reverse order.[Meye82; Nati77]

Fig. 1


Input
Initial Permutation
L_0
R_0
K(1)
g
⊕
1ST Round Output:
L(1) = R(0)
R(1) = L(0) ⊕ $g_{K(1)}$ R(0)
16TH Round Output
L(16) = R(15)
R(16) = L(15) ⊕ $g_{K(16)}$ R(15)
Inverse Initial Permutation
Output

G. RIVEST-SHAMIR-ADELMAN (RSA) ALGORITHM

The RSA algorithm is a public-key algorithm and block cipher. The algorithm is based on the difficulty of factoring large composite numbers, which is a NP-complete problem.

The cipher keys are mathematically derived using two large prime numbers, p and q. The prime numbers must meet the following properties to insure adequate security of the algorithm:

1. P and q must be at least 100 decimal digits in length.

2. The length of p and q should differ by a few digits.

3. The greatest common divisor (GCD) of (p-1) and (q-1) should be small.

4. Both (p-1) and (q-1) should have large prime factors.

5. P = ap' + 1 and q = bq' + 1, where p' and q' are large prime numbers and a and b are small prime numbers.

6. Choose p' and q' such that p' = a'p" + 1 and q' = b'q" + 1, where p" and q" are large prime numbers.

A large SK is chosen such that the GCD with SK and the product of the primes, minus one, is one:

$$GCD(SK,(p-1)(q-1)) = 1$$

((p-1)(q-1) is known as the Euler Phi Function of p & q)

PK is then derived by finding a number that is a multiplicative inverse of PK modulo the Euler Phi function:

$$PK*SK = 1 \bmod ((p-1)(q-1))$$

The ciphering function first breaks the data into blocks of 2 or more characters and assigns integer values to each

block. Next, the block is raised to the power of a cipher key and has the modulus taken with respect to p*q = N. Thus encrypting can be expressed as:

$$C = P^{PK} \bmod N$$

and decrypting as:

$$P = C^{SK} \bmod N$$

The multiplication is commutative, thus enciphering/deciphering operations are equivalent to deciphering/enciphering:

$$(X^{SK})^{PK} = (X^{PK})^{SK} = X^{SK*PK} = X \bmod N$$
(for any X between 0 & N-1)

Note, the RSA algorithm can be used as a stream cipher, but it is used as a conventional algorithm. That is, the sender and receiver use the same function and same key.[Meye82; Katz77; Rive78]

H. COMMAND AND CONTROL (C2) SECURITY SYSTEM

CSSCS operations consist of gathering information at a battalion level to forward to a support battalion. The support battalion then formats the data for entry into the C2 systems. Next, the data is transferred to division and corps level for decision processing. Lastly, the data is transferred to all the other C2 systems in ACCS for further investigation. The data will be stored in two relational databases: 1) the message database for creating, formatting and sending of raw data; 2) C2 database for the repository command and control information. The data will be transferred over any of the following type of transmission media:

commercial communication systems, wire tactical satellite, FM radio, Mobile Subscriber Equipment, or multichannel. Due to the sensitivity of the data in ACCS, there is a need for some type of security system.

The basic idea behind a security system is the protection of information stored in a computer and communicated from one computer to another, against unauthorized access. The protection mechanisms in a military application are hierarchical in nature. The hierarchy consists of issuing sensitivity levels (unclassified - top secret) to the information and classifications (uncleared - multiple category) to users. Thus, users must gain entry to the system and then have an appropriate classification to access information.

The Department of Defense (DoD) has the following requirements for a controlled multilevel security system:

1. User classification can vary.

2. The security system must handle multilevel security up to the secret level.

3. The system must handle the reference monitor requirement.

4. The code must be kept to a minimum.

5. Recovery procedures are required.

6. A security administrator is required.

7. Channel security must be addressed.

8. Audit mechanisms must include signal security-relevant events.

9. Secure information must follow established information flow.

10. The security system must be able to implement security in a network or stand alone mode.

11. The security mechanisms must be able to be transferable between heterogeneous machines.

12. The security system must be able to transfer security between the C2 and message database.

13. The security system must account for different levels of security based on physical locations.[Heng87]

An informal model has been presented for CSSCS that uses a modified Bell-Lapadula security system design. This design uses a security kernel system that incorporates an access control mechanism for subject (user, process, etc) *identification/authentication* and for object (file, process, etc) *request authorization*. The informal model also presents integrated measures for data security, through the use of a cryptosystem when transmitting and encryption techniques when storing the data.

The system will be implemented on the Tactical Army Combat Service Support Computer System (TACCS). TACCS is composed of 1,024-kilobyte RAM, keyboard and terminal screen; a 67-megabyte hard disk unit; a 630-kilobyte floppy disk unit; a 24-megabyte magnetic tape unit, and a variable-speed dot matrix printer. It can be carried to the field in three cases that together weigh 282 pounds.[Tice87]

CHOOSING A CRYPTOSYSTEM

The need for a cryptographic systems to insure security in the transmission and storage of data originates from the underlying hardware. Specifically, the communication (wires, satellite, FM radio and multichannel devises) and storage (files, disks, directory) hardware/software is not under physical control of the system users. Basic types of attacks on transmitted data include tapping into the lines, electronically listing to transmissions, or altering the data during a transmission. Attacks against stored data include copying, modifying, deleting or in the case of "*hackers*"[Land85] browsing through the system for personal gain. Thus, cryptography is employed to make the data unintelligible until it is properly decrypted. Note, that cryptography does not prevent data from being altered during transmission or deleted while stored. These problems are usually solved by the security aspects of the operating system. The following section will investigate various aspects of cryptographic algorithms that are used today.

A. BLOCK vs. STREAM CIPHERS

Stream ciphers have the ability to use the entire preceding portion of a message and the key to encrypt the current bits of the message. Block ciphers encrypt fixed sized blocks of the message independently of preceding blocks. Thus, stream ciphers can be constructed that are stronger with the respect to scrambling the data.

This strength of stream ciphers is also a drawback when used with computers. For example, if the beginning of a large file needs to be updated then the entire file must be encrypted again to insure the decryption of the file. Thus, when storing data, block ciphers enable easier decryption, updating and re-encrypting of files. Stream ciphers have analogous problems when transmitting the data, in that the transmission process must be reset to the state preceding the incorrect portion of the message.[Pope79]

Two major problems inherent to block ciphers are padding of small blocks and providing mechanisms for masking repeated patterns in the input stream. The first problem will deteriorate the throughput of the ciphering functions, obviously due to the lengthening of the message. The second also deteriorate the throughput of the function, but it will also make the implementation of the function more complex.[Meye82]

Stream ciphers are usually implemented in hardware using either feedback shift registers or J-K flip-flops. Block ciphers generally require both hardware and software in their implementation. The hardware consists of S-boxes and P-boxes and software used for breaking the messages into blocks and padding short blocks.[Lemp79]

B. CONVENTIONAL Vs. PUBLIC-KEY

The obvious difference is conventional algorithms are symmetric (same key) where public-key algorithms are

asymmetric(different keys).[Simm79] This difference has a large impact on issues of key generation and key management. The key generation of a conventional algorithm randomly selects the keys in a straight forward way, since knowledge of the enciphering key is equivalent to knowledge of the deciphering key and visa versa. In contrast, public-key algorithm the relationship between the public and private keys is purposely made obscure. Thus, a special procedure is needed to compute the public and private keys efficiently.[Meye82] In the case of key management, public-key algorithms are much simpler, since the public keys can be distributed easily.[Poke79] But, this distribution of the public keys has the disadvantage of putting more design restrictions on the algorithms since more information is given to attackers. Key management issues will be investigated further later in this section.

An important difference is that public-key algorithms are easily described mathematically and rely on the assumption that a particular, known mathematical problem is difficult to solve. But, conventional algorithm's mathematical equations describing its operation are so complex that for all practical purposes it is not possible to solve them using analytical methods. Thus, the attack disciplines of public-key algorithms are few and fixed by the algorithm's mathematical description, where conventional algorithms have the freedom to ensure that many attack disciplines are required.

Also, in conventional algorithms the deciphering process can be determined if the enciphering key is known, since all the steps in the enciphering process can easily be retraced to obtain the deciphering process. This retracing must not be possible in public-key algorithms, the encipher and deciphering paths must be independent.[Meye82]

Although there are several differences between the two algorithms, public-key algorithms are variations of conventional algorithms. Thus, the two are equivalent with respect to key distribution and digital signature. Also, in a comparison of protocols, both where found to be strikingly similar.[Need78]

C. ALGORITHMS

The only conventional algorithm that has been fully developed has been the DES algorithm, as outlined in the PRIMER. No significant changes have been added to the original DES algorithm since it's publication in 1977. There have been proposals to strengthen DES using concepts of *block chaining* with either *variable keys* or *cipher text feedback*[Meye82], but with little success.

There has been quite a bit of development and research on public-key algorithms. The RSA algorithm (also outlined in the PRIMER) has had to be further developed due to several successful attacks against it's security. The first was based on the factorization of N, the product of the two primes. By placing the 4th & 5th properties, mentioned in the PRIMER, on

the selection of the primes has made this type of attack virtually impossible using today's technology. The development of parallel processing computers has made factoring large composite numbers some what easier. In fact, 50-digit and 67-digit primes have already been factored efficiently and it is assumed that 100-digit primes will soon be factored (note that the factoring of 200-digit primes is still virtually an impossible task in polynomial time).[Wage84; Simm77] The second attack on the RSA algorithm was based on the fact that for some K:

$$P^K = 1 \bmod N$$

Thus, the message can be recovered by repeated multiplications. But, by placing property 6 (also mentioned in the PRIMER) on the selection of the primes, this attack is also avoided.

The Merkle-Hellman public-key algorithm is based on the classical NP-complete problem in number theory known as the *Knapsack* problem. The algorithm is described as follows:

> Let $A = (a_1, a_2, \ldots, a_n)$ be the secret vector of integers & $X = (x_1, x_2, \ldots, x_n)$ be the message vector of binary digits.
>
> Enciphering: $Y = A * X$ (where * is the dot product of the vectors)
>
> Deciphering: If Y_n is less than a_n, then x_n is set equal to 0 and
>
> Y_{n-1} is set equal to Y_n.
>
> Else x_n is set equal to 1 and Y_{n-1} is set equal to
>
> $Y_n - a_n$.
>
> This is repeated for each Y until all of X is recovered. (A full description can be found in [Merk78])

There have been a number of constraints (*trapdoors*) added to the basic description to make the algorithm more secure. But, it has been shown recently that even with these further conditions that the algorithm is insecure.[Adel84]

A public-key algorithm based on the undecided word problem for groups, specifically *finitely presented groups*, was presented by Neal Wagner in 1984. The algorithm consists of:

A set of generators $x_1, x_2, \ldots, x_n$

A set of relaters $r_1 = e, r_2 = e, \ldots, r_m = e$
(e = identity)

An operation: concatenation.

Replacement rules:
1) replace $x_i x_j^{-1}$ or $x_j^{-1} x_i$ by e
2) introduce $x_i x_j^{-1}$ or $x_j^{-1} x_i$
3) replace r_j or r_j^{-1} by e
4) introduce r_j or r_j^{-1}

Public key:
1) 2^N non-equivalent words $W_1, \ldots, W_N$
2) relaters $R_1 = e, R_2 = e, \ldots, R_m = e$

Private key: relaters $S_1 = e, S_2 = e, \ldots, S_k = e$

A word is a finite string and a group G is the set of equivalence classes of all possible words, where a word v is equivalent to a word w if v can be transformed into w by a finite sequence of the replacement rules. The encrypting function uses the replacement rules and R-relaters on W_i to produce the ciphered word. The decrypting function uses the S-relaters to determine which W_i the cipher word is equivalent. The S-relaters are chosen in a manner that makes it easy to determine whether two words are equivalent, but this is not so with the R-relaters alone. Although this algorithm has not been broken, Wagner presented it for

theoretical studies and its use with computer security has not been fully investigated.[Wagn84a]

John Cade developed a public-key algorithm based on permutations described as a polynomial function over the finite field $GF(2^N)$ performed on blocks of n (i.e. 110) binary digits. The enciphering permutation is a composition of three polynomial functions: P = SMT. The three polynomial functions are fixed, invertible, and have the following form:

$$M(x) = x^{2^c+1},\ T(x) = a_0x + a_1x^{2^c} + a_2x^{2^{2c}},\ \&$$

$$S(x) = b_0x + b_1x^{2^c} + b_2x^{2^{2c}}$$

where the coefficients a_k and b_k are elements of the field $GF(2^N)$. P is considered to be the public key and the coefficients of S and T constitute the private key which gives the information needed to determine the deciphering permutation $P^{-1} = T^{-1}M^{-1}S^{-1}$. This algorithm was presented in 1985 and was found to have weaknesses at the 1986 Proceeding of CRYPTO. Cade has restructured the algorithm since the proceedings, but it has not been determined, as of yet, if the algorithm is secure after the restructuring.[Cade85; Cade86]

A new approach to modern cryptographic algorithms uses exponentiation as a basis and it has been found that an efficient solution to discrete logarithm problems are of great significance. These algorithms are classified as public-key algorithms and the most prominent of these is D (decimal) sequences. D-sequences are obtained in expansions of fractions or irrational numbers and thus are decimal sequences

to arbitrary bases. The coding of the information is similar to the RSA with the following difference:

$$\text{RSA:} \qquad C = P^{PK} \bmod N$$

$$\text{D-sequence:} \qquad C = \frac{P^{PK} \bmod N}{N}$$

Although these algorithms seem promising they are still in the development stage.[KAK85]

In summary to public-key algorithms, there has been some investigation into whether NP-complete problems are a significant bases for public-key cryptosystems. The concern originates from the fact that so many proposed algorithms that are in this class have been broken (i.e. knapsack) and the current understanding of their difficulty only includes worst case analysis. Thus, recent research has been focused on undecided problems in groups to develop more secure algorithms (i.e. the word problem for groups).[Wagn85b]

D. KEY MANAGEMENT

Key management is the processes by which participants of a network or system obtain cipher keys in order to communicate or access information respectively. Key management is directly affected by the type of algorithm used in network usage, either conventional or public-key. Note also, that the key management for system use is significantly simpler than that of network use, since a secure channel between users is not needed in system usage.

In the case of a user attempting to access encrypted information on a system, the most appropriate solution is the

one presented by Akl and Taylor[Akl83] for a multilevel security system. The term multilevel security refers to hierarchy of users in which each user is assigned a security level (clearance or class) and that user can only access information at their level or lower. Thus, users, information, and cipher keys can be viewed as partially ordered sets in which users, information and keys are divided into a number of disjoint sets:

$$\text{User} = \{U_1, \ldots, U_n\}$$
$$\text{Information} = \{x_1, \ldots, x_n\}$$
$$\text{Cipher Keys} = \{K_1, \ldots, K_n\}$$

The system is then described as follows:

1) A central authority generates n cipher keys for use with a public-key algorithm.

2) For $i = 1,2,\ldots, n$, key K_i is distributed to all users in U_i who keep it secret.

3) For $i,j = 1,\ldots,n$ all users in U_j also obtain K_i if $U_i <= U_j$.

This proposed scheme has the advantage that only one copy of information is stored and the operations of enciphering and deciphering are performed once. Also, each user only needs to obtain one key and can compute all other keys needed from that key if $U_i <= U_j$. This is made possible by assigning each class of users an integer t_i so that $U_i <= U_j$ if and only if t_i evenly divides t_j; and a one-way function with the property that $f_{m \cdot z} = f_m * f_z$ (where m & z are integers and * is composition). Thus, K_i is obtained from K_j by:

$$K_i = [K_j]^{t_i/t_j} \bmod N \quad (N \text{ is the product of two primes})$$

This scheme has been well researched and proven as appropriate protection in both commercial and military implementations. There have not been any significant schemes developed for conventional algorithms for system usage.

In networks, participants must first obtain matching keys in order to communicate securely. This matched pair of keys forms a logical channel which is independent of all other logical channels and is as real as any channel created by a network's transmission protocols. Thus, the main issue in network key management is the distribution of the cipher keys to both participants wishing to communicate.[Pope79]

Conventional algorithms must distribute both cipher keys over the same insecure channel that the actual data is transferred. This creates a circular problem that is resolved by a prior distribution of a small number of keys by secure means. Initially, a host machine or set of machines are designated as a key distribution center (KDC) and a pair of matched keys are distributed to the KDC & each potential participant. Thus, once one of the participants needs to communicate securely with another, it sends a short message to the KDC over the prearranged secure channel (which usually carries a low quantity of traffic) requesting another set of keys be sent to both parties. If the KDC's protection policy permits the connection, the keys and status information is sent. A new secure logical channel is then established between the two participants independent of the KDC and

communication is started. Variations of the KDC topology include:

> *Centralized*: single KDC that all participants must access for cipher keys.
>
> *Fully Distributed*: every "intelligent node in the network is a KDC.
>
> *Hierarchical*: local, regional and global KDCs.

In each topology interconnection between networks with differing encryption disciplines is accomplished through a gateway, which is analogous to that of network gateways. The use of a particular topology is implementation dependant.

Public-key management schemes are similar to conventional in that a central authority (which publicly gives out it's encrypting key) is needed and each participant of the network needs to initially obtain a private key. But, to establish a secure logical channel between two participants, the central authority only sends one key over any channel since it is a public key of the receiving participant. An example format for establishing a secure logical channel begins with *A* sending a message to the central authority requesting communication with *B*. The authority replies with *B*'s public key encrypted using the authority's private key. *A* then decrypts the message using the authority's public key and establishes a secure channel with *B* that is independent of all other channels (*B* must also perform a similar action to obtain *A*'s current public key). There have been proposals on ways to avoid the initial request such as "phone book", caching, and certificates, but all were found to actually add to the

overhead needed for key management. Thus the overhead needed by public-key and conventional key management schemes is very similar.

E. AUTHENTICATION

Authentication is the process of identifying one member of a network communication channel to the other in a reliable and unforgeable way. The problem arises when a secret password must be sent in order to establish a communication channel between two participants. The first member that sends the password is exposed and the other member may be an imposter, who now has received the necessary information to pose to other nodes as the first member. The general solution used by most schemes [Boot81; Need78] incorporates the encryption of a rapidly changing unique value (usually the time of day or time stamp) using a prearranged key. The format of the scheme is as follows (it is assumed that both *A* & *B* have obtained matching authentication keys and a request for communication has been sent by *A*):

1) *B* sends *A* an unciphered, random, unique data item.

2) *A* encrypts the data item using a matching authentication key and sends back the data item.

3) *B* decrypts *A*'s message using its own matching key and compares it to what *B* originally sent. Thus, if they match *B* is satisfied that *A* was the originator of the message.

Note, the reliability of the authentication scheme depends on the security of the authentication keys. Thus, key management

plays a important role in authentication schemes, since both parties must be able to obtain the authentication keys securely. Authentication of system users does not incorporate encryption schemes and is usually performed by access control mechanisms of the operating system [Heng87].

F. DIGITAL SIGNATURES

Digital signatures are used for validating events such as bank transactions, contract negotiations, and military command and control orders. Signatures are an extra level of security to insure that a particular message was received from the correct source and the source is liable for that message. Using a public-key algorithm, a signature is obtained by encrypting a message using a private key and sending to a second party that has the public key to decrypt the message. The private key essentially becomes the owner's signature on the message, since the private key for all practical purposes belongs to only one person. Signatures can also be obtained using conventional algorithms if keys are assigned in a rigorous fashion (one per person).

The above approach has the problem that the author of a signed message can disavow and repudiate his signature by publicly releasing his private key and making all previously signed messages invalid. To prevent the problem a third party is introduced in which all signature messages must pass through before being sent. In a public-key algorithm the third party is known as a *notary-public*. The notary public

will time stamp the encrypted message, encipher message a second time using a separated private key, and send it back to the owner, who will use the secure channel established with the receiver to send the message. The receiver then decrypts the message using the public key of the notary-public first and then the author's.

In conventional algorithms a similar approach is taken with the use of a *network-registry* (NR). The NR are software/hardware units in the network that provide signatures. An author authenticates with a local NR, provides a message and recipient identifier. The NR times stamps and encrypts the message to form signature block before sending the message to the receiver. Once the signed message is received the recipient verifies the signature block with the NR. Several notary-public and NRs are independently established in a network with each possessing different keys. Senders are advised to request that several of the notary-public or NRs be used to insure that several failures or compromises must occur before signature validity is lost. As with authentication, protection of the keys and safe distribution is crucial.[Pope79; Boot81; Need78]

G. MINIMIZATION OF CENTRAL MECHANISMS, TRUSTED CODE & COMMUNICATION

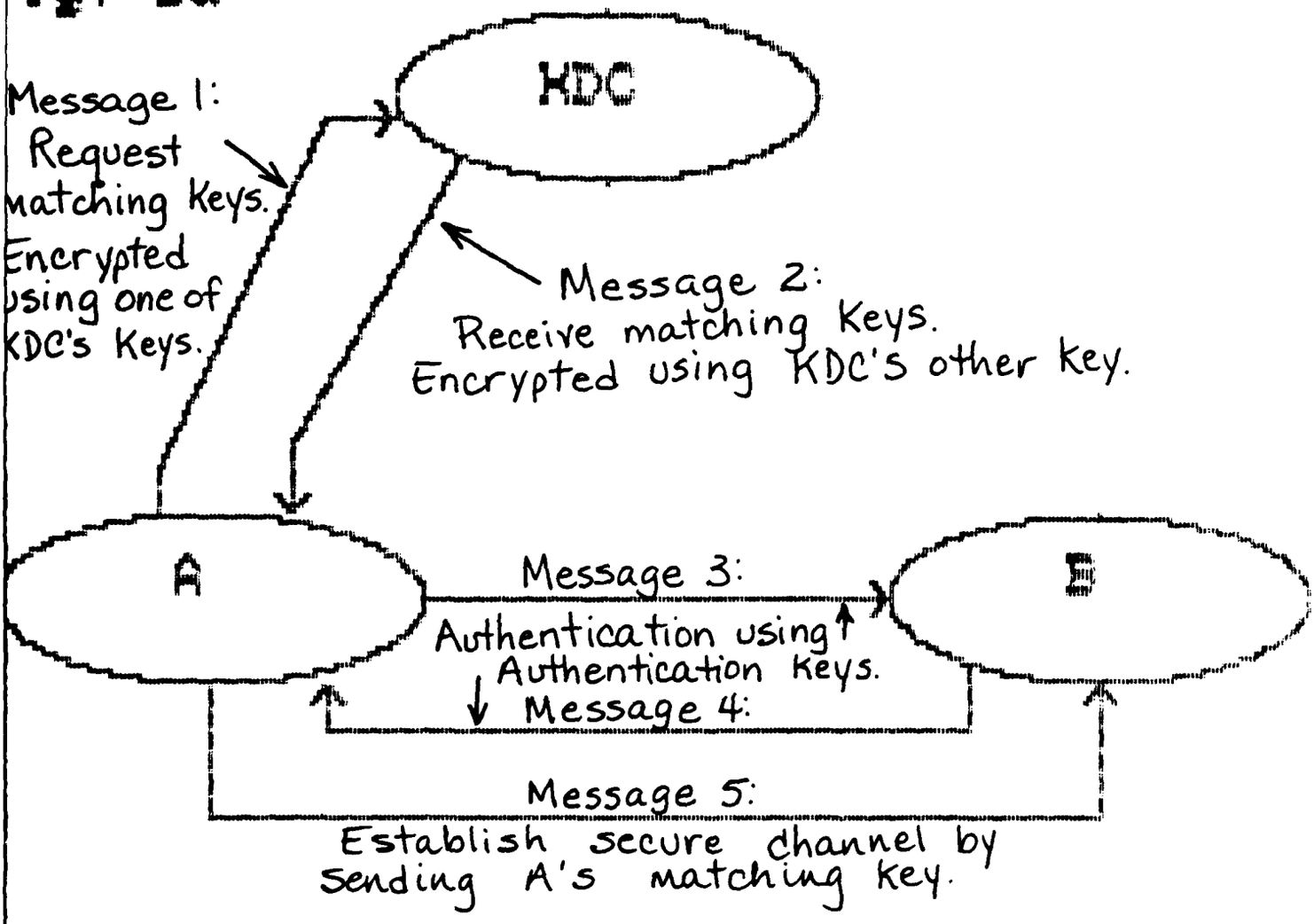
The best approach to security is to minimize the number of ways in which security can be breached from outside as well as inside the system. Possible areas to minimize are central authorities, number of messages needed to establish a secure

channel, the amount of information retained by all participants of the network, and the limitation of any single user's power over the system (superusers). Although the amount of minimization of any of the areas is implementation dependent, some concepts are presented.

The main issue concerning central authority minimization is limiting the number of them in a large network. There is a general distrustfulness that a single centralized governmental communication, or even a large common carrier, can ensure privacy and other related characteristics. Thus, a large single authority is not an appropriate solution. The military has broken it's system into five primary systems with each divided down into three subsystems as described in the Introduction. Note, that the minimization of central authorities directly limits the amount of cipher keys, since the central authority handles all key management. Also, splitting up the authority helps prevent any one user from having total control and accessibility over the system.[Heng87; Pope79]

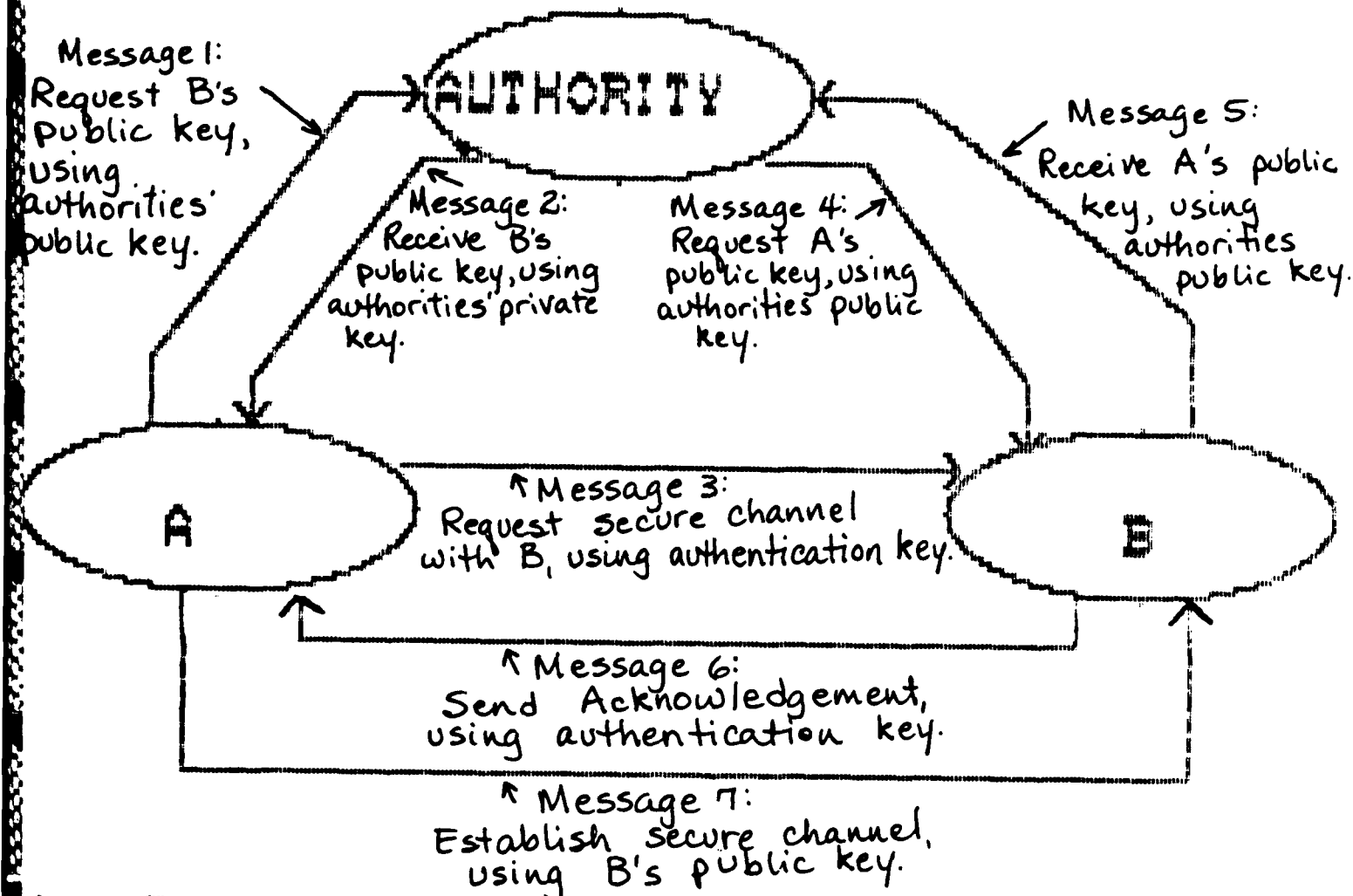
The number of messages needed for a secure channel establishment totals 5 and 7 for conventional and public-key algorithms respectively. Figure 2 depicts the message transfer for both algorithms. The number of messages can be reduced to 3 for both conventional and public-key algorithms by the use of local cache memory to store frequently used keys. Also, for public-key algorithms, certificates containing a user/public key and certifying information can be

Fig. 2a



Key Distribution for Conventional Alg.

Fig. 2b



Key Distribution for Public-Key Alg.

used to eliminate one message issued by the central authority. The authority sends the certificates out periodically (when updating is needed or after a set time) to each node to be stored as a signed message. Once a node needs to communicate, it sends the certificate to the second party. The validity of the message is then checked with the certifying information and the public key is retrieved. A major problem associated with both caching and certificates is updating invalid or obsolete keys. A global search would be needed whenever a key has been compromised, which could be costly and time consuming. Also, both must have some means of securely and correctly storing the keys, which could add to the overall message overhead. Note, that both caching and certificates would still require an internal authentication mechanism.[Pope79]

Reduction of information retained by nodes in the network can be accomplished by maintaining the databases and cipher keys (as stated earlier) at only the central authorities. An obvious drawback to this approach would be the amount of time needed to first find the correct information needed and then routing it through the network to the appropriate node requesting the information.

H. ERROR DETECTION

Noise created either by atmospheric conditions or intentional human intervention will create uncontrollable errors in a transmitted message. Prevention schemes have been

developed to both detect and correct these errors by using extra bits known as either redundancy or parity check bits. These bits possess some relation to the information being sent. For example, the Vertical or Longitudinal Redundancy Checks (VRC & LRC respectively) sum the number of ones in each message or block, and depending on the parity (odd or even) an extra bit is added to insure the parity of each message is consistent. That is, if an odd parity is used and the number of ones in the message is odd a 0 is the extra bit else another 1 is add to make the number of ones odd (even parity is analogous). These are simple detection schemes and can only detect a few errors.

A more complex method widely used in data communication is the Cyclic Redundancy Check (CRC). CRC incorporates a generating polynomial which is converted into a binary vector with ones in each position of the coefficients and zero everywhere else (i.e. $x^4 + x^2 + 1 \Rightarrow 10101$). The message has zeros appended to the low-order positions, with the number of zeros corresponding to the highest degree of the generating polynomial. Once the zeros have been appended to the message, the generating polynomial, in binary form, is divided into the message using shifts and exclusive OR's. The remainder, known as the Block Check Character (BCC) is appended to the original message before it is transmitted. The receiver strips off the BCC and divides it by the generating polynomial. If there is no remainder, the message is assumed to have no errors during the transmission. The generating polynomial for a CRC-16

scheme is $x^{16}+x^{15}+x^2+1$ and the types of errors that can be detected are summarized as follows (g is the highest degree of the generating polynomial):

> Single bit, two bit, odd number of bits in error and error burst of length $< g + 1$ have a 100% probability of detection.
>
> Error bursts of length $= g + 1$ have a $1-(1/2)^{g-1}$ probability of detection.
>
> Error bursts of length $> g + 1$ have a $1-(1/2)^{g}$ probability of detection.[Lane85]

Schemes have been developed that not only detect errors, but also correct some errors. These codes are based on finite field (Galois fields or GF) arithmetic and are classified as either block or cyclic codes.[Blah84] The codes are composed of a collection of binary code words of some fixed block length of n bits of which k bits are information bits and r=n-k redundancy bits. The value k/n provides a measure of the information content of the code words and is called the rate of the code. The codes are designed by using a generator polynomial over GF(2^m) (a Galois field with 2^m elements). Note, the generator polynomial used here and in CRC are mathematically derived to enable the codes to detect or correct specific types of errors, this is known as the design capability of the code. If the number of errors is greater then this design capability the code may fail.

The redundancy bits are calculated either by multiplying the information bits with a generating matrix or by using the information bits in a set of equations that incorporate an exclusive OR function. If equations are used when finding the

redundancy bits, to check the message for errors, the redundancy bits are tested to see if they still satisfy the equations on a pass or fail basis. Patterns can be formed from the failures that are used to locate and correct specific bits. In the case of finding the redundancy bits by using a generating matrix, the "decoding" process is based on the minimum distance between two distinct code words, known as a decoding sphere. Using the decoding spheres, syndromes (a type of discrete Fourier coefficient) are first calculated for possible erroneous code words. From the syndromes two special polynomials are constructed: an error locator polynomial whose roots are reciprocals of the locations of bits in error; and the error corrector polynomial which gives the values to be added to the erroneous bits to get the corrected message.

Specific block or cyclic codes and design information are given in the following table (parameters are given in bits):

Code Name	Parameters	Usage	Capability
Hamming	$n=2^t-1$ $k=2^t-t-1$ $r=t$	Packet Radio	Corrects single errors
Golay	n=23 k=12 r=11	Navy teletype	Corrects any pattern of <= 3 errors per 23 bits
BCH Bose-Chaudhuri-Hocquenghem	$n=2^t-1$ $k=2^t-et-1$ r=et	Very large semi-conductor memory that simulates a magnetic disk	Corrects any pattern of <= e errors per block
RS Reed-Solomon (Parameters in bytes)	$n=2^m-1$ $k=2^m-2t-1$ r=2t	Satellite links, deep space links, international terrestrial links	Corrects any pattern of <= t byte errors per (2^m-1) byte

The detection schemes will be used on the data after it has been encrypted and detection/correction of errors performed before the data is decrypted. Thus, security of the data is maintained between the ciphering devises and the error coding devices. All the schemes mentioned in this section are usually implemented in hardware to provide the best timing results in performing the coding process. Also, most of the block/cyclic codes can be implemented in parallel hardware such as pipelines or systolic system for even faster processing.[Blah84]

THEORETICAL IMPLEMENTATION

This section will first present some of the parameters and major functions of CSSCS. Followed by a possible configuration and implementation of a cryptographic scheme integrated with CSSCS's structure and operating system.

A. SYSTEM DETAILS

CSSCS can be described as a loosely coupled occasionally connected network of independently operating computers. The system is considered occasionally connected due to movement of units on the battlefield and the possible loss of entire units/computers as the ba tle progresses; of course even in peace time the system will lose units due to either movement or mechanical failures. The computers (TACCS hardware) will be tied together using the Signal Corps devices mentioned in the primer. The cryptographic implementation outlined later will give a general description that should be functional with any of the data communication devices.

The computers will be issued to corps, division, support battalion and battalion levels. Users will include enlisted personnel who may handle data input and report generation, through senior commanders who will use the output of the various support packages to make command and control decisions. There are sixty-two types of reports under six major headings and it has been estimated that up to 850 transaction per day will be processed during a wartime situation.

There are two types of information flow in CSSCS, internal flow within CSSCS and transfer of information with the other nodes of ACCS. The internal flow will be collected at battalion level and placed into a report or Joint Interoperability of Tactical Command and Control Systems (JINTACCS) format to be forwarded to the Division Support Command (DISCOM). For example, a poisonous chemical has been located by field personnel at the battalion level. Chemical corp personnel then prepare a report and send it to their support battalion, who enter the information into CSSCS in an appropriate format. The information can then be manipulated at the DISCOM or COSCOM (Corps Support Command) to make corrections to battle plans. Thus, raw data reports of battle situations flows in an upward fashion from battalions to corps and battle plan corrections in a downward fashion.

The second type of information flow mention passes information in and out of CSSCS to other nodes of ACCS, by way of either DISCOM or COSCOM. For example, a similar chemical

downwind report is prepared and sent to COSCOM, but it was sent by either the Fire Support, Air Defense, Electronic Warfare, or Maneuver Control nodes of ACCS rather then a subordinate unit of COSCOM. The information is then evaluated and change to plans are sent back down to subordinate units in the same manner as before. Information collected by CSSCS not pertaining to their operations can also be forwarded on to the appropriate node of ACCS in a similar manner as just mentioned. Appendix A gives a listing of reports that are received and transmitted by CSSCS.

The operations of CSSCS from a user view at each level is viewed as follows. User at the support battalion and lower level will be able to update, send, and receive files; create, send, and receive reports; as well as use the C2 database. Although the majority of the time task performed at this level will be in updating file, creating, sending and receiving reports. DISCOM users have similar capabilities with the addition of aggregating the lower reports into higher level reports and access to the other nodes of ACCS. COSCOM users will have exactly the same capabilities as DISCOM, but the data collection is in a more aggregated form and COSCOM will provide an interface with the echelons and above corps (EAC) units. All users at every level must login into the system and pass identification/authentication procedures, as well as access control mechanisms when requesting to perform some operation. These security measures will be performed by the operating system security functions.

The proposed design is based around the cryptographic solution to a multilevel security problem[Akl83] (presented in the algorithm subsection of the previous section) that will use the modified RSA algorithm presented by Williams for the encrypting and decrypting functions.[Will80] Since the number of users in CSSCS will be large, the improved key generation algorithm purposed by MacKinnon and Akl will also be incorporated. MacKinnon and Akl proved that the original algorithm had the problem that the t_i given to the users, in order to compute all other cipher keys needed, can get quite big even for a small number of users (i.e. 20 users and worst case $t_i = 28 \times 10^{25}$).[MacK83] See appendix B for the key generation and management schemes as presented by Akl, Taylor, and MacKinnon.

There will be three levels of central authorities (CA), EAC will act as CA for COSCOM, COSCOM in turn for DISCOM, and DISCOM for support battalions & lower levels. The rest of the design will address the relation of EAC & COSCOM and how information will be ciphered between/at those levels. The design description will be very similar for each of the other two CAs and their subordinate nodes, differences will be pointed out when necessary.

B. KEY GENERATION

Initially, when the system is first configured, EAC will generate two large primes, p & q that will satisfy the RSA properties, to compute M = p*q, which is used to compute the first deciphering key, K^d_1. A third prime, N, is also

generated by EAC (and made public for the calculation of subordinate key of a level) that is larger then M (M < N), which is used to compute the first encipher key, K^{e}_{1}, that will be assigned to the highest security level. The military has partitioned the clearance levels for its systems in the following manner (these would be considered the ordered poset (partially ordered set), (U_1, U_2,...,U_n), for Akl algorithms):

Uncleared: lowest level of security.

Not cleared but authorized access to sensitive unclassified information.

Confidential

Secret

Top Secret/Current Background Investigation

Top Secret/Current Special Background Investigation

One Category

Multiple Category: highest security level.

Thus, the first cipher key pair is issued to users at the Multiple Category clearance level.

Next, a enciphering and decipher key pair is calculated and assigned to each of the other security levels using M, the first pair of cipher keys, a set of prime numbers (with the size of the set equal to the number of security levels), and an unique integer. This is accomplished by first assigning each clearance level, U_i, a prime number, p_i, and an unique

integer, t_i, in the following way (U_j is considered the level just above U_i):

> If U_i is the highest level then assign the lowest prime, 2, and set t_i equal to 1.
>
> If level U_i is equal or lower then U_j, assign U_i the prime assigned to U_j. Note, in CSSCS there is not more then one clearance type on the same level, but if there where the second clearance would get the next lowest prime in the set, the third the next prime, etc.
>
> Once all of the primes have been assigned, go through the clearance levels in the same order assigning unique integers, t_i, such that:
>
> $$t_i = \prod_{U_j \not\geq U_i} p_j$$
>
> where $U_j \not\geq U_i$ means all levels not accessible (higher) then U_i.

The enciphering and deciphering keys at each level can then be calculated using the following function:

$$K_i = [K_j]^{t_i/t_j} \pmod{M}$$

where K_j is the current level and K_i is the next level down. Also, K^e would be used when calculating the next enciphering key and K^d for the deciphering key. The actual encipher and deciphering would be performed exactly the same way as described when the RSA algorithm was discussed earlier.

As an example of the whole process, let figure 3 represent the Hasse diagram of the poset of clearances, outlined above, with the top node representing the multiple category clearance level (labeled U_7) and the bottom node uncleared level (labeled U_1). Also, let p = 61, q =47, M=61*47 = 2867, N = 2903 and N - 1 = 2902. The numbers inside

Fig. 8

Hasse
Diagram of
A poset.

of the nodes of the Hasse diagram are the unique integer assigned to that node and the number on the outside is the assigned prime. The first node, U_7, was assigned a 1 as the unique integer and the first prime 2 since it was the highest clearance level. The second node, U_6, was also assigned 2 as its prime since the first node can access the second, and since the only node unaccessible to U_6 is U_7, U_6 is assigned 2 its unique integer. U_5 was assigned 2 as its prime for the same reason that U_6 was assigned 2; U_5 is assigned 4 as its unique integer since both U_6 & U_7 are both unaccessible to U_5 and the product of U_6's & U_7's primes is 4. Thus, a pattern is formed and it should be obvious how the rest of the nodes where assigned integers and primes.

Next, as depicted by the RSA algorithm, K^d is chosen so that it has a greatest common divisor (GCD) of one with the Euler Phi function of p and q, picking $K^d_7 = 167$ satisfies this requirement. K^e is then chosen so $K^e * K^d = 1 \bmod (N-1)$, and $K^e_7 = 921$ satisfies this requirement. Both key values can be computed and checked for correctness by incorporating the Euclidean Algorithm for finding GCDs and multiplicative inverses. Finally, the rest of enciphering and deciphering keys are computed using the first pair of keys and the information contained in the Hasse diagram as follows:

$$K^d_6 = [K^d_7]^{t_6/t_7} \bmod (N-1) = 167^{2/1} \bmod 2902 = 1771$$
$$K^e_6 = [K^e_7]^{t_6/t_7} \bmod (N-1) = 921^{2/1} \bmod 2902 = 857$$

$$K^d_5 = 1771^{4/2} \bmod 2902 = 2281 \text{ or } 167^{4/1} \bmod 2902 = 2881$$
$$K^e_5 = 857^{4/2} \bmod 2902 = 243 \quad \text{or } 921^{4/1} \bmod 2902 = 243$$

$$K^{d}_{4} = 2281^{8/4} \bmod 2902 = 1771^{8/2} \bmod 2902 = 167^{8/1} \bmod 2902 = 2577$$
$$K^{e}_{4} = 243^{8/4} \bmod 2902 = 857^{8/2} \bmod 2902 = 921^{8/1} \bmod 2902 = 1009$$

.
.
.

If the deciphering key computed (by raising the previous key to some power and mod it with N-1) for the next level was not relatively prime to N-1; then Akl and Taylor include an algorithm for computing a unique integer which depends on the previous key and N-1 (the algorithm is included in appendix B).

Since the military's clearance levels are totally ordered, it is possible for the users with the highest level clearance to compute the keys of all of the other levels with its one key. This is a convenient mechanisms, but it creates a few problems with respect to security. If the set of keys just produced were used for all resources of the system, then a similar situation as with single superusers arises. Specifically, if a high security user gives out the deciphering key (or an intruder obtains it), the whole system would be vulnerable and everything in the system would have to be re-enciphered. It would be better to use different sets of keys for each resource and for each subordinate of a CA. But, having too many sets causes users to maintain a lot of keys, which is a contradiction with one of the design principles of the cryptographic solution to multilevel security, mainly limiting the number of keys maintained by users. Thus, the EAC will generate two sets of keys for each COSCOM that is a

subordinate, one for database usage when ciphering stored data, and the other for communication. Note, that the problem of having to re-encipher all information inside the domain of a set of keys any time any key in that set is compromised, is unavoidable.

Both COSCOM and DISCOM would perform the exact same process for generating keys, the only difference is that a different set of parameters and keys would be generated. This scheme will help make sure the compromise of keys at one unit level (COSCOM, DISCOM, etc) does not compromise information at all the other levels and possibly other ACCS systems.

C. KEY DISTRIBUTION

Key distribution for CSSCS will be performed similarly at EAC to COSCOM and COSCOM to DISCOM, but differently for DISCOM to support battalions & lower levels. Since EAC and COSCOM are typically at stationary and secure locations (compared to battalions who are normally close to the front line and mobile), cipher keys can be distributed from EAC to COSCOM (and COSCOM to DISCOM) by some type of courier service. For example, the keys (as well as the unique integers and prime number N) could be placed on a floppy disk that contains a protection mechanism that requires a legitimate user to enter some type of identification or the disk reformats itself, which would destroy any information on the disk. The disk is then hand carried from EAC to COSCOM, with no real threat if it is intercepted by an intruder while in route.

The battalion computers will be housed in lightweight, nuclear-hardened mobile shelters (new generation of jeeps) currently being built by Grumman Aircraft Systems. It is envisioned that these units will be deployed from DISCOM to the field for a period of a few weeks. Thus, before deployment the cipher keys and other relevant information can be stored in the system. It may seem inappropriate to put all the keys in a system so vulnerable to attack, but recall that battalion level operations are mainly gather information and receiving specific orders. Therefore, capabilities of the system are quite limited compared to DISCOM (or COSCOM) and the keys used at each battalion level would be a completely different set than those used at other levels or other battalions.

Keys will be distributed to users of a clearance level similarly to objects (classifications and compartments associated with each key) and only deciphering keys will be distributed. When a user initially needs to obtain a cipher key (has not previously obtained one), the object the key will be used on will be compared with the user by the access control mechanisms of the operating system. If the user passes, then the access rights of the key and user are compared in a similar manner. If both access right checks are passed the key is issued to the user in a secure manner. A user already possessing a key will still have to pass both access right checks, and if he doesn't then the operating system will deny deciphering of the object.

Enciphering will be performed by the operating system after a request by a user and if the sensitivity of the information is high enough to merit ciphering. For example, using the chemical downwind message, the personnel at battalion level would try to access the communication processes and ask for the message to be sent to DISCOM. The operating system would verify access rights to the communication processes, and determine if the message should be encrypted and at what level of clearance. If the requests are granted, the message is enciphered using an enciphering key that was generated at DISCOM, who has the original copy of the deciphering key for the same level. DISCOM can decipher the message upon receiving it and determine if the message should be forwarded to COSCOM or stored. In either case the message would be re-enciphered (if needed) using the cipher keys generated by COSCOM. If the message is forwarded to COSCOM and if the message was store at COSCOM, it would be ciphered using the cipher keys generated by EAC.

D. AUTHENTICATION AND DIGITAL SIGNATURES

Authentication would have to be performed in different ways: between CA and one of its subordinates; between two subordinates of the same CA; between two COSCOMS (or DISCOM) level units of different ACCS systems. Authentication with the CA and a subordinate is directly accomplished when the set of keys is generated for that specific subordinate. As soon as the keys have been loaded into the subordinate, a logical channel is established with the CA for every clearance level

and communication can occur immediately (provided the users have passed all access rights placed on the communication processes).

Authentication between subordinates of the same CA would be accomplished in similar manner as depicted in figure 2b. Using the appropriate enciphering key, the initiating subordinate would make a request to the CA for a pair of keys to be generated for use in a logical channel between the subordinates. The CA would respond by returning the enciphering key used to communicate with the second subordinate encrypted using the initiating subordinate's appropriate key. The initiating subordinated would then send a request (using the newly require key) to the second subordinate for the establishment of a secure channel be set up between the subordinates. The second subordinate would request the deciphering key for the initiating subordinate (using the appropriate key) from the CA (who encrypts the message in a similar manner as before). After receiving the key the second subordinate would establish the channel with the first. It may be possible to allow the subordinates to maintain those keys for some set time or until compromised, at which time new keys would have to be obtained before communication could resume.

The third type of authentication would require the use of a gateway. Since different cipher keys or even algorithms are used within different ACCS system, the gateway would be situated between the two systems and would have to receive a

message for one system, decipher it, and re-encipher the message for the next system. The gateway would have to contain the specific algorithms used by the two systems it is servicing and it would have to be included when cipher keys are generated and distributed. The actual authentication is accomplished in a manner similar to the first type described above. As soon as the gateway is given the cipher keys from both systems, a logical channel is established. More than one gateway will service the same system and each COSCOM and DISCOM of the systems will distribute keys to several of the gateways. This helps to prevent systems from being disconnected from each other in case a gateway is compromised. But, again it is assumed that the security at the DISCOM and COSCOM unit level is stable.

Digital signatures will be accomplished by having the CAs act as notary-publics. As above there will be three different situation that will be handle some what differently. The CA will notarize all messages sent to it by its subordinates, and the CA will notarize messages for two subordinates. In the case of intersystem communication, the gateway will act as the notary-public.

E. MISCELLANEOUS

A block of text would first be broken down into some set of sub-blocks. If 200 digit prime numbers where used, the sub-blocks could be as big a 199 characters, but it would be more conventional to keep the size of the sub-block to some multiple of the size of the lines in the messages (i.e. if the

message lines are 80 characters, then sub-block could be 160 characters long). Each sub-block would then be enciphered using appropriate keys and either put into the appropriate format for storing or put into some format corresponding to the protocol used for the particular data communication device. For stored data, error correction would be perform after encipherment and appended onto the end of the text block. The protocol format may break the blocks down further, embed the ciphered text between starting/trailing messages/codes used by the communication hardware/software, and perform some type of error correction (whose result is usually part of the trailing message/codes information).

The topology of the system (number of COSCOMs, DISCOM, etc and how they would be arranged) would depend on the terrain and situation of the battle. The use of graph theory would be used to help compute the configurations of several different types of scenarios.

F. HARDWARE AND SOFTWARE

The software of the cryptographic design will consist of the RSA algorithm, key generation and distribution procedures, data communication processes (protocols, interfaces, etc), and notary-public processes. The RSA algorithm source code is about 36K in size and takes approximately 4 to 5 CPU seconds (the algorithm was implemented on a HP-3000) to encipher (and decipher) 160 characters of text. The characters where enciphered in 80 character blocks using 150 digit prime numbers. The actual size and time specifications of the key

generation/distribution and notary-public software is not known at this time. The data communication software has been developed (tested and implemented) by the Signal Corps of the Army, information about the Signal Corps systems was not available. All of the software should be maintained in the security kernel of the operating system inorder to insure that it will not be easily accessible and tamper proof.

The hardware needed by the cryptographic design will consist of an error correction device (usually a modified FIR filter and shift registers), floppy disk for cipher key distribution, and all necessary network hardware (interfaces, communication lines and device, etc). Although any of the error correction schemes mentioned earlier would be appropriate for this application, it seems that the Reed-Solomon codes have been incorporated into most of today's applications (satellite links in deep space, CD disks both musical and store applications, etc). The floppy disk is standard issue on TACCS hardware and will be available to systems. Again, Signal Corps has already developed the hardware of the network.

CONCLUSIONS

The major conclusion, as far as cryptographic algorithms and their parameters are concerned, is number of algorithms that seem secure and the direction in which development of algorithms is taking. Only RSA and DES have been used in any application and all of the implementations use them in a block mode of ciphering. Since, only two algorithms have not been

broken and most of the algorithms (including those two) are classified as NP-complete problem, it is thought that stronger more reliable algorithms will be found in the class of undecided problems.

The use of the cryptographic solution to a multilevel security problem as the basis of the design proved to be quite adequate for CSSCS applications. The design should add very little overhead to the operating system in both time and storage. Only a few additional procedures would have to be added to the security kernel since CSSCS is already based on a multilevel access control mechanism. One drawback of the design is that the additions would have to be made to every EAC, COSCOM, and DISCOM. Although the design presented was a very broad description, its purpose was to show that the cryptographic solution could be used for the ciphering of transmitted and stored data in CSSCS.

It is apparent that no system can be made totally secure using today's technology and computer architecture, unless everything in the systems is monitored all of the time. But, this would cut down on the speed and usefulness of computers system, since half of the CPU time would be devoted to monitoring itself. Thus, every type of security scheme will only absolutely protect some percentage of the information and will have a no protection at all with the other percentage. The major weakness of the present cryptographic algorithm is blind luck on the part of an intruder in decoding intercepted encrypted messages and the amount of overhead needed when a

key has been compromised (the re-encipherment of anything ciphered that used the key, was derived from the key, or the key was derived from).

Further work would examine the detailed description, simulation of the design and full implementation of the system.

REFERENCES

[Akl83] Akl, S.G. & Taylor, P.D., "Cryptographic Solution to a Multilevel Security Problem," Advances in Cryptography: Proceedings of CRYPTO 82, Plemun, New York, 1983.

[Blah84] Blahut, Richard E., Theory and Practice of Error Control Codes, Addison-Wesley Publishing Company Inc., Reading, Mass., 1984.

[Boot81] Booth, Kellogg S., "Authentication of Signatures Using Public Key Encryption," Commun. ACM, Vol. 24, No. 11, Nov 81, pp. 770-772.

[Cade85] Cade, J.J., "A New Public-Key Cipher Which Allows Signatures," Second SIAM Conference on Applied Linear Algebra, April 1985.

[Cade86] Cade, J.J., "A Modification of a Broken Public-Key Cipher," Proceedings of CRYPTO 86.

[Heng87] Hengst, Paul T., "Security Requirements and Informal Security Model for the Combat Service Support Control System," Naval Postgraduate School, March 87.

[Kak85] Kak, Subhash C., "Encryption and Error-Correction Coding Using D Sequences," IEEE Trans. on Computers, Vol. c-34, No. 9, Sept. 1985.

[Katz77] Katzan, Harry, The Standard Data Encryption Algorithm, Petrocelli Books Inc., 1977.

[Land85] Landreth, Bill, Out of the Inner Circle: A Hacker's Guide to Computer Security, Microsoft Press, Feb. 1985.

[Lane85] Lane, Malcolm G., Data Communications Software Design, Boyd & Fraser Publishing Company, Bostan, 1985.

[Lemp79] Lempel, Abraham, "Cryptology in Transition," *ACM Computing Surveys*, Vol. 11, No. 4, Dec. 79, pp. 285-304.

[Merk78] Merkle, R. & Hellman, M.E., "Hiding Information and Signatures in Trapdoor Knapsacks," IEEE Trans. on Inform. Theory, Vol. IT-24, No. 5, pp. 524-530, Sept. 1978.

[Meye82] Meyer, Carl H. & Matyas, Stephen M., Cryptography: A New Dimension in Computer Data Security, John Wiley & Sons, Kingston New York, 1982.

[Nati77] National Bureau of Standards, Data Encryption Standard, Federal Information Processing Standards Publication, Jan. 77.

[Need78] Needham, Roger M., Shcroeder, Michael D., "Using Encryption for Authentication in Large Networks of Computers," Commun. ACM, Vol. 17, No. 17, Aug. 74, pp. 437-442.

[Pope79] Popek, G.J. & Kline, C.S., "Encryption and Secure Computer Networks," *Commun. ACM*, Vol. 11, No. 4, Dec. 79, pp 331-358.

[Rive78] Rivest, R. L. , Shamir, A. & Adelman, L.M., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Commun. ACM*, Vol. 21, No. 2, Feb. 78, pp. 120-126.

[Simm77] Simmons, G.J. & Norris, M.J., "Preliminary Comments on the M.I.T Public-Key Cryptosystem," CRYPTOLOGIA, 1977, pp. 406-414.

[Simm79] Simmons, Gustavus, "Symmetric and Asymmetric Encryption," *ACM Computing Surveys*, Vol. 11, No. 4., Dec. 79, pp. 305-330.

[Tice87] Tice, Jim, "TACCS to Connect Units to Single Data Base," *Army Times*, Feb. 2, 87, pp. 29-30.

[Wagn84a] Wagner, Neal R., "A Public-Key Cryptosystem Based on the Word Problem," Advances in Cryptology: Proceedings of CRYPTO 84, ed. by G.R. Blakely and D. Chaum.

[Wagn84b] Wagner, Neal R., "Searching for Public-Key Cryptosystems," IEEE, (incomplete citation).

[Will80] Williams, H.C., "A Modification of the RSA public-key encryption procedure," IEEE Trans. Inform. Theory IT26:726 (1980).

APPENDIX A

ACCS REPORTS

The Army Command and Control System Inter-Control Elements Interfaces specifications have developed the following for cross referencing report exchange among the five nodes of ACCS view from the CSSCS system (only the first two of the six major titles are presented to provide the general idea to the reader):

Nodes:
- MC = Maneuver Control Nodes
- AD = Air Defense
- EW = Electronic Warfare
- FS = Fire Support

Functions:
- T = Node can transmit report
- R = Node can receive report

Information Unit Title	MC	AD	EW	FS
1. Enemy Information				
Aircraft Report	TR	TR	TR	
Assessment			R	
Enemy Activity	TR	TR	TR	
Enemy Weapons Systems		TR		
Intelligence Summary			R	
2. Friendly Information				
ADM	R			
Adjacent Unit Situation		TR		
Aircraft Allocations/Priorities	R			
Aircraft Requirements	TR	R		
Ammunition Supply Rate	R	TR		T
Area of Operations		TR		
Assets Available	T	R		TR
Available Supply Rate	T			
Basic Load Percent Fill	TR	R		R
Battle Losses (Equip)	TR	R	R	R
Casualties		R	R	R
Command Controlled Items	TR	TR		T
Command Mission	R			
Communication Centers	R			

APPENDIX B

CRYPTOGRAPHIC SOLUTION TO A MULTILEVEL SECURITY PROBLEM

The following algorithm was presented in the paper by Akl and MacKinnon[MacK, 83] for controlling the size of t_i and alleviating the problem of having the users in two or more clearances collaborating to compute a key to which they are not entitled:

> Let $U_1,\ldots,U_n$ be some poset and $\leq$ be the operation associated with the poset.
>
> The integer t_i satisfies the following properties:
> 1) $t_j \setminus t_i$ if and only if $U_i \leq U_j$
> ($\setminus$ means divides evenly)
> 2) gcd $(t_j) \not\setminus t_i$ for $U_j \Rightarrow U_i$
> ($\not\setminus$ means does not divide evenly)
> 3) the t_i's are not larger than those obtained by the original algorithm

Algorithm:

Step 1: Find the top node U_k such that $U_i \leq U_k$ for all i, and assign it the prime $p_k = 2$; if no such node exists, then create a dummy node U_k satisfying the condition and let $p_k = 1$. Determine the level e_i of each node U_i.

Step 2: For e = 1 to the maximum level do:

1) Let L be an array containing the U_i's such that $e_i = e$.

2) Let P be a set containing the first n primes, i.e. P = {2,3,5,7,11,...}.

3) For each U_i in L do (in any order)
   1) Assign to U_i a prime p_i which is the smallest element of P satisfying the condition: for all U_j such that $e_j < e$ and $p_j = p_i$, $U_j \geq U_i$.

   2) P = P - $\{p_i\}$.

Step 3: For each node U_i set

$$t_i = \prod_{U_j \leq U_i} p_j$$

($\prod$ is the product of p_j)

The following was presented in the original cryptographic solution paper by Akl and Taylor[Akl83] for computing a unique positive integer relatively prime to N-1. The algorithm is only used if the deciphering key used to produce the next level's cipher key is not relatively prime to N-1 after being raised to the appropriate power and mod-ed N-1.

s = N-1 value and the deciphering key being used is r.

Algorithm:

```
relative_prime (r,s)

    if (r is even) then r = r + 1
    outer = false

    while (outer = false) do

     begin
      w = r
      inner = false

      while (inner = false) do

       begin
        x = gcd (w,s)
        if (x = 1) then inner = true
        else w = w/x
       end

      if (w ≠ 1) then outer = true
      else r = r + 2
     end

    relative_prime = w
```

BIBLIOGRAPHY

Armendariz, Efraim P. & McAdam, Stephen J., Elementary Number Theory, Macmillan Publishing Co., Inc., New York, 1980

Davies, Donald W., The Security of Data in Networks, IEEE Computer Society, Piscataway, NJ, 1981.

Denning, Dorothy, "Secure Personal Computing in an Insecure Network," Commun. ACM, Vol. 22, No. 8, Aug. 79, pp. 476-482.

Denning, Dorothy E. & Sacco Giovanni, M., "Timestamps in Key Distribution Protocols," Commun. ACM, Vol. 24, No. 8, Aug. 81, pp. 533-537.

Evans Jr., Arthur & Kantrowitz, William, "A User Authentication Not Requiring Secrecy in the Computer," Commun. ACM, Vol. 17, No. 17, Aug. 74, pp. 437-442.

Gavish, Bezalel & Hasn, Pirkul, "Computer and Database location in Distributed Computer Systems," IEEE Trans. on Computers, Vol. C-35, No. 7, Jul. 86, pp. 583-590.

Gifford, David K., "Cryptographic Sealing for Information Secrecy and Authentication," Commun. ACM, Vol. 25, No. 4, Apr. 82, pp. 274-286.

Lamport, Leslie, "Password Authentication with Insecure Communications," Commun. ACM, Vol. 24, No. 11, Nov. 81, pp. 770-774.

Merkle, Ralph C., "Secure Communications Over Insecure Channels," Commun. ACM, Vol. 21, No. 4, Apr. 78, pp. 294-299.

Merkle, Ralph C., "On the Security of Multiple Encryption," Commun. ACM, Vol. 24, No. 7, Jul. 81, pp. 465-467.

Voydock, Victor L. & Kent, Stephen T., "Security Mechanisms in High-level Network Protocols," ACM Computing Surveys, Vol. 15, No. 2, Jun. 83, pp. 135-171.

END

8-87

DTIC